输入和输出

1. print会依次打印每个字符串，遇到逗号“,”会输出一个空格
2. str = raw\_input()

name = raw\_input('please enter your name: ')

1. raw\_input() 直接读取控制台的输入（任何类型的输入它都可以接收）。而对于 input() ，它希望能够读取一个合法的 python 表达式，即你输入字符串的时候必须使用引号将它括起来。

数据类型和变量

1. Python可以处理任意大小的整数，支持大数计算，无限位数。
2. Python还允许用r''表示''内部的字符串默认不转义
3. True False and or not None

字符串和编码

1. Unicode把所有语言都统一到一套编码里，最常用的是用两个字节表示一个字符（如果要用到非常偏僻的字符，就需要4个字节）。UTF-8编码把一个Unicode字符根据不同的数字大小编码成1-6个字节，常用的英文字母被编码成1个字节，汉字通常是3个字节，只有很生僻的字符才会被编码成4-6个字节。
2. 普通的字符串在Python内部都是ASCII编码的

|  |
| --- |
| >>> ord('A')  65  >>> chr(65)  'A' |

以Unicode表示的字符串用u'...'表示

|  |
| --- |
| >>> print u'中文'  中文  >>> u'中'  u'\u4e2d' |

把u'xxx'转换为UTF-8编码的'xxx'用encode('utf-8')方法

# -\*- coding: utf-8 -\*-

1. print 'Age: %s. Gender: %s' % (25, True)

list和tuple

1. list的方法

|  |
| --- |
| >>> classmates = ['Michael', 'Bob', 'Tracy']  >>> len(classmates)  3  >>> classmates[-1]  'Tracy'  >>> classmates.append('Adam') #添加  >>> classmates  ['Michael', 'Bob', 'Tracy', 'Adam']  >>> classmates.insert(1, 'Jack') #插入  >>> classmates  ['Michael', 'Jack', 'Bob', 'Tracy', 'Adam']  >>> classmates.pop() #删除末尾  'Adam'  >>> classmates  ['Michael', 'Jack', 'Bob', 'Tracy']  >>> classmates.pop(1) #删除位置  'Jack'  >>> classmates  ['Michael', 'Bob', 'Tracy']  >>> classmates[1] = 'Sarah' #替换  >>> classmates  ['Michael', 'Sarah', 'Tracy'] |

1. list里面的元素的数据类型也可以不同

|  |
| --- |
| >>> L = ['Apple', 123, True] |

1. tuple，元组，一旦初始化就不能修改。

只有1个元素的tuple定义时必须加一个逗号,，来消除歧义：

|  |
| --- |
| >>> t = (1,) |

条件判断和循环

1. 只要x是非零数值、非空字符串、非空list等，就判断为True，否则为False。
2. 循环

|  |
| --- |
| for i in range(n):  ###  for i in list:  ###  for i,v in enumerate(s):  ### |

使用dict和set

1. dict, key-value存储方式,map。dict的key必须是**不可变对象**。

|  |
| --- |
| >>> d = {'Michael': 95, 'Bob': 75, 'Tracy': 85}  >>> d['Michael'] #通过键获取值  95  >>> d['Jack'] = 90 #通过键赋值  >>> d['Jack']  90  >>> 'Thomas' in d #判断键是否存在  False  >>> d.get('Thomas') #不存在键，返回None  >>> d.get('Thomas', -1) #指定不存在键的返回值为-1  -1  >>> d.pop('Bob') #通过键删除键值对  75  >>> d  {'Michael': 95, 'Tracy': 85} |

和list比较，dict有以下几个特点：

1. 查找和插入的速度极快，不会随着key的增加而增加；
2. 需要占用大量的内存，内存浪费多。

而list相反：

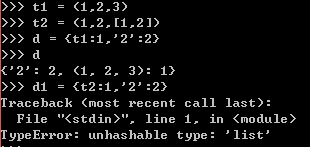
1. 查找和插入的时间随着元素的增加而增加；
2. 占用空间小，浪费内存很少。

所以，dict是用空间来换取时间的一种方法。dict内部存放的顺序和key放入的顺序是没有关系的。

1. set是一组key的集合，但不存储value，key不重复，无顺序。

|  |
| --- |
| >>> s = set([1, 2, 3]) #创建注意不要忘了set  >>> s.add(4) #添加  >>> s  set([1, 2, 3, 4])  >>> s.remove(4) #删除  >>> s  set([1, 2, 3])  >>> s1 = set([1, 2, 3])  >>> s2 = set([2, 3, 4])  >>> s1 & s2 #交集  set([2, 3])  >>> s1 | s2 #并集  set([1, 2, 3, 4]) |

1. 对于不变对象来说，调用对象自身的任意方法，也不会改变该对象自身的内容。相反，这些方法会创建新的对象并返回，这样，就保证了不可变对象本身永远是不可变的。



调用函数

1. 函数名其实就是指向一个函数对象的引用，完全可以把函数名赋给一个变量，相当于给这个函数起了一个“别名”：

|  |
| --- |
| >>> a = abs # 变量a指向abs函数  >>> a(-1) # 所以也可以通过a调用abs函数  1  >>>abs=1  >>>abs(-111)  Traceback (most recent call last):  File "<stdin>", line 1, in <module>  TypeError: 'int' object is not callable  >>>del abs |

定义函数

1. 空函数

|  |
| --- |
| def nop():  pass  if age >= 18:  pass |

调用函数

1. 内置函数isinstance实现参数类型检查

|  |
| --- |
| def my\_abs(x):  if not isinstance(x, (int, float)):  raise TypeError('bad operand type')  if x >= 0:  return x  else:  return -x |

1. 可以返回多个值，return x,y。返回的是一个tuple。

函数的参数

1. 设置默认参数

|  |
| --- |
| def power(x, n=2):  s = 1  while n > 0:  n = n - 1  s = s \* x  return s |